2.13 TOPOGRAPHIC CHARACTERISTICS

Topographic effects in mission-level models are effects due to terrain or due to other natural or man-made features. Man-made features include buildings, bridges, machinery, etc., while natural features include forests, rivers, swamps, etc. Although topographic features could affect many aspects of a scenario (e.g., movement or decision-making), the effects discussed in this section are their effects on energy transmission; i.e., clutter, multipath, diffraction, and line of sight masking. Suppressor does not model multipath or diffraction, but it does simulate clutter and masking effects for sensing, communicating, and jamming.

Suppressor uses processed DMA DTED to determine terrain masking effects. Any of the standard DMA terrain data base formats can be read and processed by the DMA step and then further processed by the EDB step. If the DMA terrain ‘squares’ used are at different longitudinal densities, all of them must be processed at the lowest density (measured in seconds of arc) in the scenario. (This would apply, for example, if the scenario included terrain both above and below 50 degrees latitude.)

The DMA step converts the terrain data from points to (flat) triangular areas that adjoin one another to produce a continuous but non-differentiable surface. All triangles used are approximately equilateral with sides of length equal to 2n times the distance between terrain points. By selecting a minimum and a maximum value of n (greater than or equal to one), the user may select triangles of different sizes to account for varying terrain smoothness; i.e., the model will create smaller triangles whenever the first triangle created gives an altitude at a DMA latitude and longitude that is unacceptably different (from the user’s perspective) from the DMA altitude there. The smallest triangle available has sides with length equal to two times the distance between DMA points.

2.13.1 Functional Element Design Requirements

The design requirements for the topographic characteristics functional element are:

a. The model will provide a capability to input clutter power as a function of range and sensor elevation angle for each radar sensor.

b. Masking due to the curvature of the earth will be computed whenever the user supplies an effective earth radius. When specified, masking will affect energy transmission for sensing, communicating, and jamming.

c. Masking due to terrain will be computed whenever the user supplies DMA terrain data. When terrain is defined, masking will affect sensing, communicating, and jamming.

2.13.2 Functional Element Design Approach

Design Element 13-1: Clutter

Clutter is caused by radar signals which bounce off the ground or sea surface and interfere with the signals which are reflected directly from the target to the receiver. The best clutter algorithms include a model of the terrain with its surface features, and the amount of clutter varies depending on the type of terrain surface. Suppressor currently includes a

CLUTTER-TABLE data item which is used to define a clutter template around each radar site. The CLUTTER-TABLE data item currently is not terrain specific.

The Suppressor TDB input file allows a CLUTTER-TABLE to be associated with each radar type. The table has dimensions for target altitude and range between the radar and target, and includes clutter power values which degrade the radar signal-to-noise computation.

Design Element 13-2: Masking

Suppressor includes two algorithms which represent the effects of masking. The first is horizon masking which prevents long range sensors from detecting low targets because of the earth's horizon. The second is terrain masking which can interfere with sensors detecting targets, communications between two sites, or jamming, due to intervening terrain blocking the line of sight.

The horizon masking algorithm is based on the EFFECTIVE-EARTH-RADIUS TDB data item. Every sensor receiver in Suppressor may include the EFFECTIVE-EARTH-RADIUS data item. Most Suppressor analysts use four-thirds of the earth's radius for radar sensors, and six-fifths of the radius for optical and infrared sensors. If this data item is present, then every sense event for that sensor type includes a test for horizon masking. This is the horizon masking test derived from *Radar System Design and Analysis* by A. Hovanessian, and performed in Subroutine LOSCHK:

Determine the distance from the target to the horizon:

$$D_t = \sqrt{A_t^2 + 2A_tR} \qquad [2.13\text{-}1]$$

then determine the distance from the sensor to the horizon:

$$D_s = \sqrt{A_s^2 + 2A_sR} \qquad [2.13\text{-}2]$$

If the 3D-distance between the sensor and target is greater than the sum of the two distances to the horizon, then the sensor and target are masked by the horizon, i.e., Masked if $D_t + D_s < D_{3d}$.

The variables used in Eqs. [2.13-1] and [2.13-2] are defined as:

A_s	Altitude of the sensor antenna above the terrain (meters)
A_t	Altitude of the target above the terrain (meters)
D_s	Distance from the sensor to the horizon (meters)
D_t	Distance from the target to the horizon (meters)
D_{3d}	Distance between target and sensor (meters)
R	Effective earth radius (meters)

The terrain masking algorithm makes use of an optional triangular faceted terrain model. Suppressor includes the preprocessing step which allows users to create a triangular terrain model from Digital Terrain Elevation Data (DTED) provided by the Defense Mapping

Agency. The triangular terrain model is optional with any Suppressor simulation. If the model is included, clear line of sight is required for every successful sensing; if the terrain model is omitted, terrain masking is ignored.

The terrain masking algorithm traverses the line of sight between the sensor and target. At every point where the line of sight crosses an edge of a terrain triangle, the altitude of the line of sight is compared with the altitude of the terrain. If the terrain is higher, then masking is assumed and the algorithm stops. If the terrain is lower, then the algorithm continues until the line of sight is traversed.

2.13.3 Functional Element Software Design

Clutter Module Design

Subroutine OBSRDR is used to compute and evaluate the radar range equation for each radar sense event. The code to look up a clutter power from the CLUTTER-TABLE is only a subset of the module. A complete design for Subroutine OBSRDR is in Functional Element 2.5.3 Radiance / Transmittance in this template. The design for the clutter subset is given here:

```
*begin logic to perform radar sensor calculations:
   *when within sensing limits:
      *look up time of sense chance;
      *look up target position pointer;
      *look up receiver position pointer;
      *look up receiver data pointers;
      *calculate range from sensor to target;
      *invoke logic to calculate jammer signals;
      *calculate square of range from xmtr to target;
      *calculate range to target;
      *look up peak transmitter power out;
      *adjust power for pulse doppler, pulse compression
      *invoke logic to target signal calculations;
      *when clutter is modeled:
         *perform lookup to determine clutter power;
         *when terrain data:
            *invoke logic to get terrain altitude under target;
         *end of test for terrain.
      *end of test for clutter.
      *look up internal receiver noise;
      *evaluate visibility for target vs clutter plus noise;
       .
       .
       .
      *when target is visible over clutter+noise and jamming:
         *target is visible;
       .
       .
       .
   *end of test for within sensing limits.
*end of logic for OBSRDR.
```

Masking Module Design

Subroutine LOSCHK performs the horizon masking algorithm, and if the sensor and target are not masked by the horizon, invokes Subroutine TRILOS to determine if the terrain model could prevent an interaction between the sensor and target. This is the top-level design for Subroutine LOSCHK:

```
ABSTRACT     check line of sight between two objects
   *begin logic to check line of sight between two objects:
      *when curved earth defined:
         *set source and target altitude;
         *when terrain data:
            *invoke logic to get source altitude of surface in MSL;
            *invoke logic to get target altitude of surface in MSL;
         *end of test for AGL altitude.
         *when above the terrain:
            *when target farther away than grazing angle:
               *radar horizon from S.A. Hovanessian;
               *set flag to no line of sight if target below horizon;
            *end of test for grazing applies.
         *end of test if above the terrain:
      *end of test for flat or curved surface.
      *when line of sight and terrain:
         *when terrain masking edges exists:
            *invoke logic to see if los still crosses mask edge;
            *when an intersection occurs:
               *set flag based on masking presence;
            *end of test for intersection occurring.
         *end of test for terrain masking edge.
         *invoke logic to masking due to terrain;
      *otherwise, did not check terrain:
         *set edge pointers to signify no checking this time;
      *end of test for terrain.
   *end of logic for LOSCHK.
```

Subroutine TRILOS is the module which performs the terrain line-of-site masking test. This is a complex algorithm which traverses the triangular faceted terrain model between the sensor and a target, searching for a triangle edge which is high enough to mask the line of site. This is the top-level design for Subroutine TRILOS:

```
ABSTRACT     determine if line of sight exists across terrain
   *begin logic to determine if line of sight exists across terrain:
      *set flag that line of sight exists (until contradicted);
      *initialize values assuming target is higher;
      *reset values when target is lower;
      *look up coordinates of end points;
      *when either point below highest terrain, lower within terrain:
         *calculate range vector from first to last point;
         *determine distance and components of range vector;
         *when tallest point lower than stopping point:
            *compute shorter distance to search;
         *set up the offset to the first terrain point;
         *narrow down the choices for the exit edge;
            *reset the index for LRITE's new perspective;
         *loop, while triangles are left to examine:
            *select the exit edge; INDEX will point from LRITE to
            * LLEFT until reset at the end of this loop;
               *reset the index by two counterclockwise
            *determine distance, intersected side vector;
            *when vectors are parallel:
               *set intersection to end of side;
            *otherwise, solve for intersection:
               *determine point of intersection for vectors;
            *end of test for parallel vectors.
            *when any distance left along range vector:
               *calculate altitude at exit face intersection;
               *when range vector is below this altitude:
                  *no line of sight so set the flag;
                  *set masking edge pointers and collect stats;
               *otherwise, keep searching for a mask:
                  *test for running off the edge of terrain;
                     *test for transition to a higher density;
                        *fraction from LRITE determines new edge;
                     *test for transition to lower density;
                     *move the index one clockwise;
                  *end of test for terrain boundary.
               *end of test for altitude clearance ok.
            *end of test for any distance left.
         *end of loop for triangles left.
         *if line of sight is OK, set masking edge to 0;
      *end of test for either point below highest terrain point.
   *end of logic for TRILOS.
```

2.13.4 Assumptions and Limitations

Only one clutter table can be defined for each sensor type, even though the scenario may include multiple sensors of the same type at different locations. For scenarios containing multiple sensors at separate sites, with each site having appreciably different clutter profiles, the calculation of clutter return at some sites may be unrealistic. This could pose a problem in target detection only for sensors lacking sophisticated clutter-rejection filters. Suppressor does not currently include a representation for multipath or diffraction.

2.13.5 Known Problems or Anomalies

When using variable resolution terrain, there is a potential infinite loop in the line-of-site algorithm, Subroutine TRILOS. This only occurs when using a terrain model with variable resolution terrain. Suppressor users can prevent this problem by using only terrain models with a constant level of resolution.